Abstract

Logic programming languages, such as Prolog, provide a high-level view of programming. These languages allow a form of programming where one declares the logic of the problem, plus the control necessary for efficient execution. Logic programs can take advantage of recent parallel architectures by exploiting implicit parallelism, where and-parallelism results from the parallel execution of goals, and or-parallelism results from the parallel execution of alternatives to the same goal.

Andorra-I is an experimental parallel Prolog system that transparently exploits both dependent and-parallelism and or-parallelism. It implements the Basic Andorra Model, a parallel execution model for logic programs in which determinate goals are executed in parallel. This model not only combines two of the most important forms of implicit parallelism in logic programs, it also allows a form of implicit coroutining. This means that Andorra-I not only supports standard Prolog but also provides the capabilities of flat committed-choice languages.

The main subject of this thesis is the Andorra-I preprocessor, that supports the execution of Prolog programs in the Basic Andorra Model. The preprocessor analyses clauses heads and builtins in the body to generate determinacy routines. These routines are used at run-time to verify when at most a clause will match a goal. To cater for Prolog programs containing builtins such as side-effects or some cuts that depend on a left-to-right order of execution, the preprocessor includes a sequencer that generates sequential annotations to guarantee the correct execution of these builtins. An abstract interpretation module performs global analysis to assist the sequencer in detecting the cuts and meta-predicates which need sequencing. Finally, the preprocessor includes a compiler that generates WAM-style code for each clause.

With the aid of the preprocessor, a number of substantial Prolog applications have already been successfully ported to Andorra-I. We discuss the performance of the different components of the preprocessor and the performance of the Andorra-I system as a whole, and finally we present some possible extensions to the Basic Andorra Model.